An Area-efficient Mixed-Precision Accelerator with Output-Error-based Quantization for ViT

Subin Park, Juhyuk Ahn, Soomin Rho, Kwangrae Kim, Ki-Seok Chung*
Department of Electronic Engineering
Hanyang University, Seoul, Korea
{subinpark, juhuyk123, smrho, kksilver91, kchung}@hanyang.ac.kr

Abstract— **Vision Transformer (ViT) has achieved remarkable performance in computer vision tasks. However, its large number of parameters poses challenges for deployment on resource-constrained devices. Mixed-precision quantization is widely used to reduce the model size. To improve accuracy while minimizing the use of high bit-width precision, selecting the appropriate precision for each tensor is crucial. In this paper, we propose a precision selection strategy that leverages the mean squared error of linear operation outputs to improve accuracy with minimal use of high bit-width tensors. Moreover, we propose a processing element that shares most of its internal resources to support mixed precision. On ViT-Base with ImageNet, our method achieves a 0.706% accuracy improvement and 1.83× speedup over a prior work with identical area constraints.**

Keywords; Vision Transformer; Mixed-Precision Quantization; Hardware Accelerator

I. INTRODUCTION

Recently, in the field of computer vision, Vision Transformer (ViT) has outperformed convolutional neural networks (CNNs). However, the large number of parameters in the ViT model makes its deployment on resource-constrained edge devices challenging. In particular, linear operations, such as query, key, and value generation, linear projection, and fully connected (FC) layers, require a large number of parameters and take up a dominant portion of the overall runtime, making it crucial to accelerate these operations.

Mixed-precision quantization has been widely adopted in edge devices. To improve accuracy and reduce hardware requirements, many SW-HW co-design studies employ custom or adaptive data types [1, 2]. Among them, ANT[1] employs mixed precision of 4 and 8 bits and proposes a hardware architecture that supports it. The quantization framework of ANT operates in two stages. First, it selects the data type for each tensor among various 4-bit data types—integer (INT), floating point (FP), flint (a mixture of INT and FP), and power-of-two (PoT). This selection is based on the lowest mean-squared error (MSE) of quantization for each data type on the tensor. After selecting the data type, the bit-width of several input pairs (weight, activation) is increased from 4-bit to 8-bit for higher accuracy. It is applied preferentially to the input pair with a higher sum of the two MSE values. However, accuracy eventually depends on minimizing the quantization error of the outputs of linear operations. Simply adding the MSEs of the two inputs, as in ANT, which corresponds to the quantization error based on the input distributions, may not minimize the actual output quantization error and may even degrade model accuracy. To support mixed-precision, ANT includes 8-bit accumulators alongside 4-bit processing elements (PEs). However, since these are active only during 8-bit operations, hardware utilization is not good. Therefore, ANT offers low computational efficiency in terms of power consumption and area, making it unsuitable for edge devices.

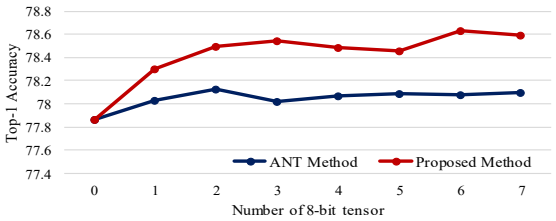

Figure 1. Accuracy result across different numbers of 8-bit quantized tensors

In this paper, we propose an output-quantization-error-based precision selection method, utilizing the MSE of the output rather than the input of linear operations to reflect the quantization error better. To confirm the effectiveness of the proposed method, we design an area-efficient accelerator that supports mixed precision. Our PE shares most of the internal resources when supporting different precision levels, enabling high hardware utilization. On the ViT-Base with the ImageNet dataset, our method achieves a 0.706% higher accuracy and a 1.83× speedup over a prior work with almost the same area.

II. PROPOSED METHOD

A. Output-Quantization-Error-Based Precision Selection

To perform linear operations, we apply both adaptive data types and mixed precision. We follow the ANT framework for data type selection while constraining the data types to INT and PoT for better hardware utilization.

In our method, the precision selection between 4-bit and 8-bit is based on the MSE of the output, not the input. First, we measure the MSE of the output quantization error by performing linear operations using activations and weights on both 32-bit FPs and quantized data types. After computing the

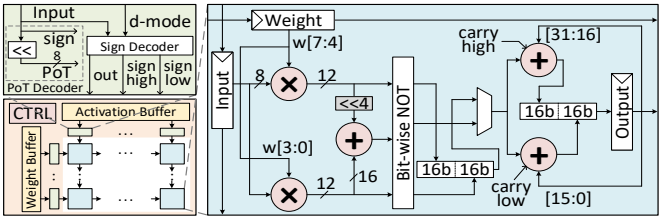

Figure 2. Proposed accelerator architecture

2025 22nd International SoC Design Conference (ISOCC) | 979-8-3315-8642-3/25/$31.00 ©2025 IEEE | DOI: 10.1109/ISOCC66390.2025.11330033

MSE of all layers, linear operations with high MSE values are quantized to 8-bit, up to the number specified by a user-defined hyperparameter. Since 8-bit PoT incurs hardware overhead, only an 8-bit integer (INT8) is used for the 8-bit quantization, regardless of the original 4-bit tensor data type.

For the ViT-Base model, we quantized 50 activation–weight tensor pairs, covering every linear operation in the model. As shown in Figure 1, applying 8-bit quantization to only about 15% of them—chosen by our output-error-based strategy—consistently outperforms ANT. Following the proposed hardware architecture in Section II.B, which favors using the smallest possible number of 8-bit tensors, we minimized 8-bit usage by evaluating from 1 to 7 tensors.

B. Accelerator Architecture

In contrast to ANT, which requires additional accumulators for 8-bit operations that are not used during 4-bit computations, resulting in low hardware utilization, we propose a PE that shares most of its internal resources to support mixed precision. To achieve this, we pack two 4-bit weights, enabling the PE to perform either a single 8-bit operation or two 4-bit operations in parallel. This structure enhances hardware utilization and is particularly effective for our method, as the proposed precision selection strategy minimizes the use of 8-bit tensors, resulting in a predominance of 4-bit operations.

Figure 2 shows the proposed accelerator architecture. Activations and weights are stored in 4-bit INT (INT4), 4-bit PoT (PoT4), or INT8 in their respective buffers. Each decoder interprets data types and precision levels based on the decode-mode (d-mode) signal, which includes four components: d-mode-0 (INT8×INT8), d-mode-1 (INT4×INT4), d-mode-2 (PoT4×INT4), d-mode-3 (PoT4×PoT4). Since INT4 weights are packed before being passed to the PE, d-mode 1 and 2 perform 8-bit×two 4-bit operations, whereas d-mode 0 and 3 execute standard 8-bit×8-bit operations.

Each decoder consists of a PoT and a sign decoder. The PoT decoder extracts a sign signal and an unsigned INT8 value, while the sign decoder outputs a sign signal and an unsigned 8-bit value (either INT8 or packed INT4), depending on the d-mode. To reduce hardware complexity, our PE utilizes two unsigned multipliers to enable sharing across all modes, with sign processing handled during accumulation. To apply sign information via carry signals of packed INT4 values, two separate sign signals (high, low) must be extracted. In d-mode 1 and 2, two 4-bit results from the multiplier are first bitwisely inverted, then the results are concatenated and accumulated into upper and lower 16-bit values based on the decoder's sign signals. In d-mode 0 and 3, which require 8-bit×8-bit operations, the results from both multipliers are summed after shifting the upper 4-bit of one operand, followed by bitwise inversion and accumulation without concatenation.

III. Experimental Result

We implement our algorithm using Pytorch with the ImageNet dataset for ViT-Base. Table I summarizes the accuracy across various data types and bit-width cases. Our method achieved 0.706% higher accuracy than ANT with the same number of 8-bit tensors (6 out of 50 pairs).

We designed our accelerator in RTL using Verilog HDL and SystemVerilog, and synthesized it with a 15nm NangateOpenCell library. For comparison, we also designed and synthesized ANT using the same technology libraries. As shown in Table II, ANT uses a 64×64 array of 4-bit PEs (equivalent to 1,024 8-bit PEs), whereas our design employs 48×64 8-bit PEs, totaling 3,072. Despite using 3× more 8-bit PEs, our design achieves comparable or smaller areas. This efficiency is attributed to our proposed PE, which shares most of its internal resources while supporting mixed precision.

For performance evaluation, we used DnnWeaver[3], a cycle-accurate simulator. For a fair comparison, latency was compared under identical conditions with ANT, including 128 bytes per cycle bandwidth, an on-chip buffer size of 512KB, and 6 input pair tensors quantized to 8-bit. As shown in Table II, our method achieves a 1.83× speedup over ANT.

TABLE I. Accuracy comparison of ViT-base on ImageNet

Scheme	Data type	Bit-width	Accuracy
Baseline	FP	32	80.968
ANT[1]	Integer	4	72.170
		8	80.582
	INT-PoT	4	77.868
		8	80.532
		4, 8†	78.08
Ours			78.632

† : Indicates mixed-precision, with 6 of 50 tensor pairs in 8-bit.

TABLE II. Area, power, and latency characteristics

Architecture	ANT[1]		Ours	
Component	Decoder	PE(Row×Column)	Decoder	PE(Row×Column)
Number	128	64×64	48, 64	48×64
Area[mm^2]	0.001	0.714	0.004	0.705
	0.715		0.709	
Power[W]	3.27		3.08	
Latency[μs]	46.455472		25.324832	

IV. Conclusion

We propose an output-quantization-error-based precision selection method for mixed precision, which determines tensor precision based on the output results. Furthermore, we propose an area-efficient accelerator to support mixed precision. On the ViT-Base model with the ImageNet dataset, our method achieves 0.706% higher accuracy and a 1.83× speedup under the same conditions of 8-bit tensor usage compared to prior work with identical area constraints.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIT) (RS-2024-00409492)

References

[1] C. Guo, et al. "Ant: Exploiting adaptive numerical data type for low-bit deep neural network quantization," in 55th Annual International Symposium on Microarchitecture (MICRO), pp. 1414-1433, 2022.

[2] C. Guo, et al. "Olive: Accelerating large language models via hardware-friendly outlier-victim pair quantization," in 50th Annual International Symposium on Computer Architecture (ISCA), pp 1-15, 2023.

[3] H. Sharma, et al. "From high-level deep neural models to fpgas," in 49th Annual International Symposium on Microarchitecture (MICRO), pp. 1–12, 2016.

979-8-3315-8642-3/25/$31.00